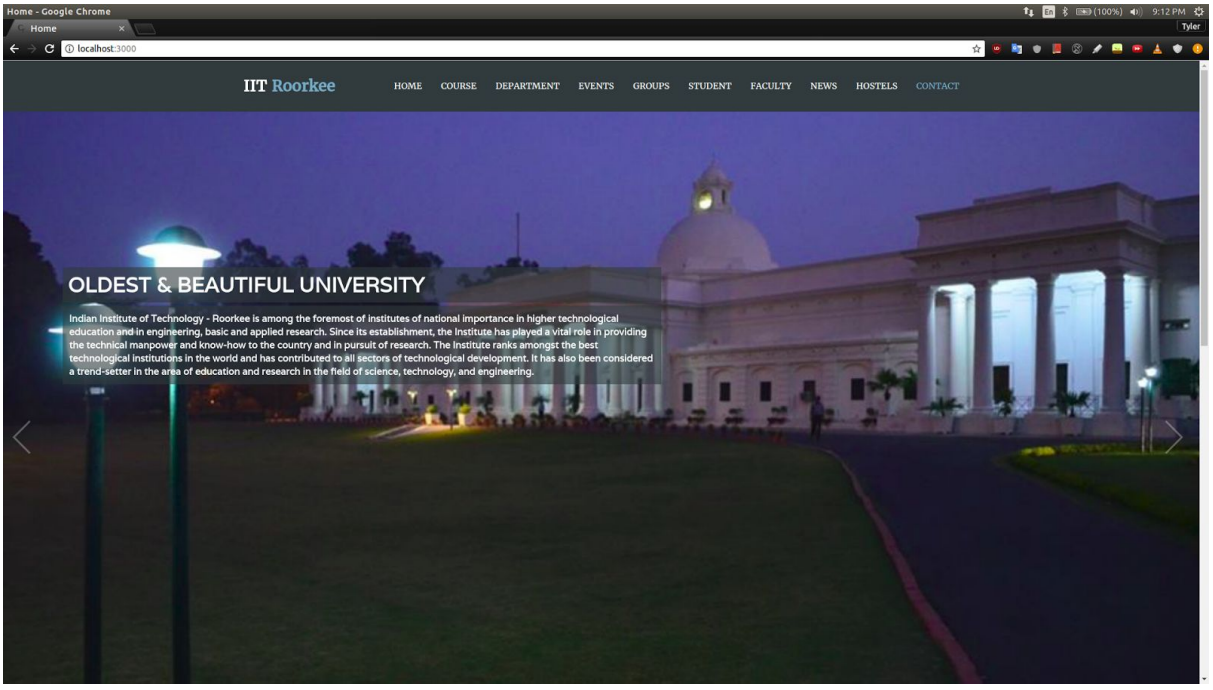
Database Management System

Course Project - Final Report

Project No: 2D

MockCollege Database

GithubLink: https://github.com/msharsha555/Classroom



HarshaVardhan MIryala(15114045)

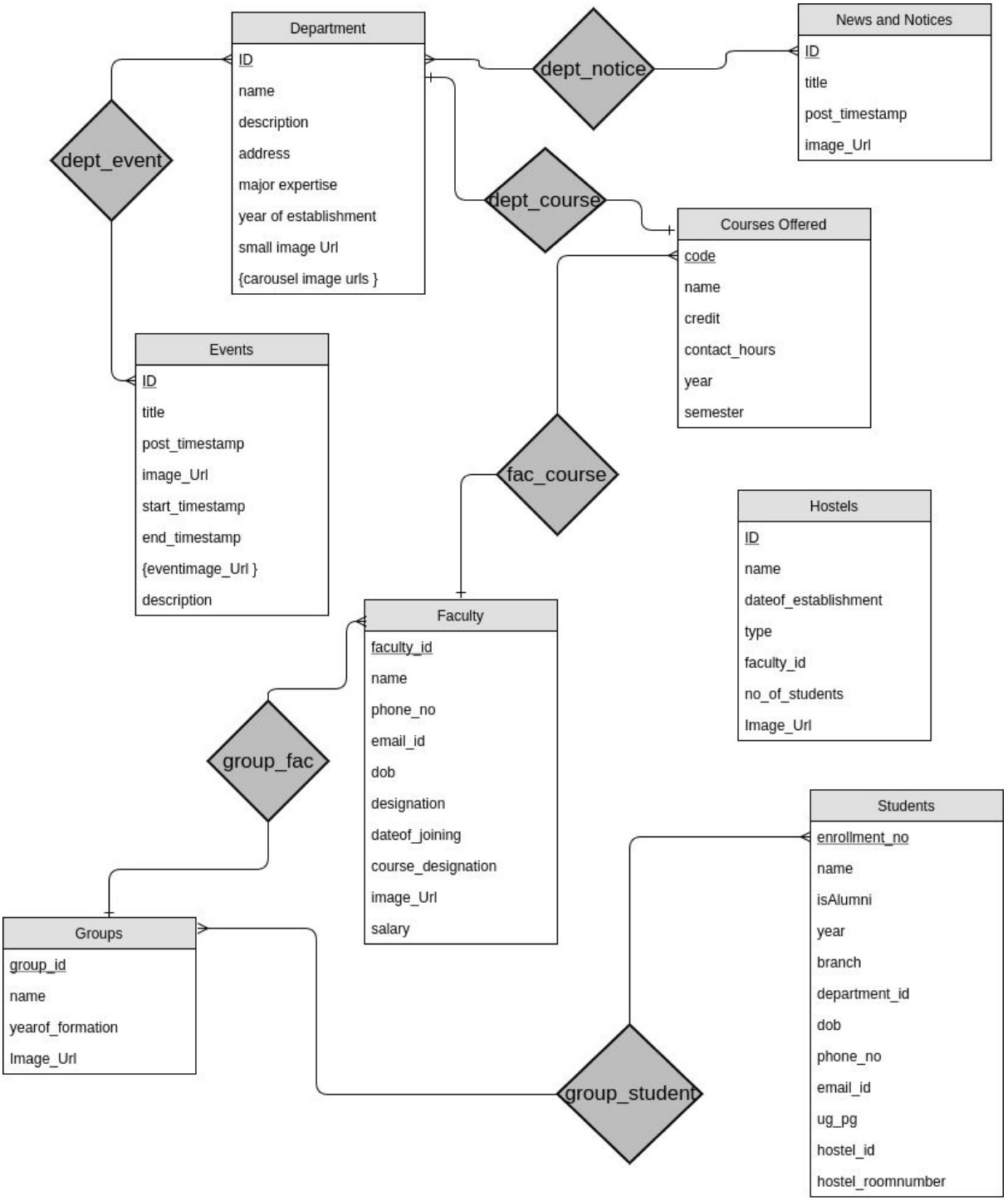
Sri Harsha Majeti (15114044)

Sanju Prabhath Reddy (15114042)

Utsav Mangal (15114075)

Siraz Sheikh (15114065)

ER Diagram :


Department
ID
name
description
address
major expertise
year of establishment
small image Url
{carousel image urls }
dept_notice
News and Notices
ID
title
post_timestamp
image_Url
dept_event
dept_course
Courses Offered
code
name
credit
contact_hours
year
semester
Events
ID
title
post_timestamp
image_Url
start_timestamp
end_timestamp
{eventimage_Url }
description
fac_course
Hostels
ID
name
dateof_establishment
type
faculty_id
no_of_students
Image_Url
Faculty
faculty_id
name
phone_no
email_id
dob
designation
dateof_joining
course_designation
image_Url
salary
group_fac
Students
enrollment_no
name
isAlumni
year
branch
department_id
dob
phone_no
email_id
ug_pg
hostel_id
hostel_roomnumber
Groups
group_id
name
yearof_formation
Image_Url
group_student

Initial Schemas :

Department: (
ID, name, description, address, major expertise, year of establishment, small image Url, {carousel image urls}
)

News and Notices : (
ID, title, post timestamp, image Url, description
)

Dept_notice: (
d_id, n_id
)

Events: (
ID, title, post timestamp,central_image Url, event start timestamp, event end timestamp,description,{event_image Url}
)

Dept_event: (
d_id, e_id
)

Courses: (
D_id, code,faculty_id, name, credit, contact hours, year, semester
)

Faculty: (
Group_id,faculty id, name, phone no., email id, dob, designation, date of Joining, course_specialization, image URI, salary
)

Groups: (
group id, name, year of formation
)

Students: (
enrollment no., name, isAlumni, year, branch, department Id, dob, phone no. Email Id, ug_pg, hostel Id, hostel room number
)

Hostel : (
Id, name, date of establishment, type,faculty Id, Number of students, Image Url
)

NOTE :

In Group entity,{student_id} is not needed if u want to represent a relationship there.Similarly for the group_id in Students. Also there is no need of representing a many to one relationship between Students and Hostel as hostel_is is already present in the entity.

The above are the obtained schemas with the corresponding primary keys.

The FDs that can be inferred from the schemas are :

Department:

- *Id -> name, description, address, major expertise, year of establishment, small image Url, {carousel image urls}*

There is no FD where the left hand side is not a primary key.ID is the primary key.

News and Notices :

- *Id -> title, post timestamp, image Url, description*

There is no FD where the left hand side is not a primary key.ID is the primary key.

Dept_event :
There is no FD where the left hand side is not a primary key.(e_id,d_id) is the primary key.All attributes form the candidate key.

Dept_notice:
There is no FD where the left hand side is not a primary key.(d_id,n_id) is the primary key.All attributes form the candidate key.

Events :

- *Id -> title, post timestamp, image Url, event start timestamp, event end timestamp* and *description*

There is no FD where the left hand side is not a primary key.ID is the primary key.

Courses:

- { D_id, code, faculty_id } -> name, credit, contact hours, year, semester

There is no FD where the left hand side is not a primary key.code is the primary key.

Faculty :

- { Group_id,faculty id } -> name, phone no., email id, dob, designation, date of Joining, course specialization, image URl, salary

There is no FD where the left hand side is not a primary key.faculty_id is primary key.

Groups :

- group id -> name, year of formation

There is no FD where the left hand side is not a primary key.group_id is the primary key.

Students:

- enrollment no. -> name, isAlumni, year, branch, department Id, dob, phone no. Email Id, ug_pg, hostel Id, hostel room number

There is no FD where the left hand side is not a primary key.enrollment_no is the primary key.

So from the above FDs we can say that only 1NF normalisation would suffice and after that all the schemas are in 5NF.

1 NF needs to be done in Department and Events.

So, the new schemas formed are

Dept_images : (
d_ID,carousel_image Url
)

Event_images : (
e_ID,event_image Url
)

The present FDs is the minimal closure and we don't need to do any furtherprocessing.

Hence , the final schemas along with the attribute initialisations are -

Department
{
ID varchar(20)
Name varchar(50)
Description varchar(2000)
Address varchar(100)
major expertise varchar(50)
year of establishment int(4)
small image Url varchar(100)
Primary key ID
}
News
{
ID varchar(20)
Title varchar(20)
Post timestamp TIMESTAMP
Description varchar(2000)
image Url varchar(100)
Primary key ID
}
Dept_notice
{
d_ID varchar(20)
n_ID varchar(20)
Primary key (d_ID,n_ID)
Foreign key d_ID references Department
Foreign key n_ID references News
}
Events
{
ID varchar(20)
Title varchar(20)

post timestamp TIMESTAMP
central_image Url varchar(100)
event start timestamp TIMESTAMP
event end timestamp TIMESTAMP
Description varchar(2000)
Primary key ID
}

Dept_event

{
e_ID varchar(20)
d_ID varchar(20)
Primary key (e_ID,d_ID)
Foreign key e_ID references Events
Foreign key d_ID references Department
}

Courses

{
d_ID varchar(20)
Code varchar(20)
Faculty_id varchar(20)
Name varchar(50)
Credit int(10)
contact hours number(6,2)
Year int(10)
Semester int(10)
Primary key (d_ID,Code,Faculty_id)
Foreign key d_ID references Department
Foreign key faculty_id references Faculty
}

Faculty

{
-- Group_id varchar(20)
faculty_id varchar(20)
Name varchar(50)
phone no. varchar(20)
email id varchar(100)
Dob TIMESTAMP
Designation varchar(2000)
date of Joining TIMESTAMP
Course_specialization varchar(20)
image URI varchar(100)

salary int(20)
Primary key (faculty_id,Group_id)
Foreign key Group_id references Group
}

Groups
{
Group_id varchar(20)
Name varchar(50)
++ Description
year of formation int(20)
Primary key group_id
++ Faculty_Id
}

Student
{
enrollment no. varchar(20)
name varchar(50)
isAlumni boolean
Year numeric(4,0)
Branch varchar(20)
department Id varchar(20)
Dob TIMESTAMP
phone no varchar(20)
Email Id varchar(100)
Ug_pg varchar(20)
hostel Id varchar(20)
hostel room number int(20)
Primary key enrollment_no
Foreign key (department_id,hostel_id) references (department,hostel)
}

Dept_images
{
d_ID varchar(20)
carousel_image Url varchar(100)
Primary key (d_ID,carousel_image)
Foreign key d_ID references Department
}

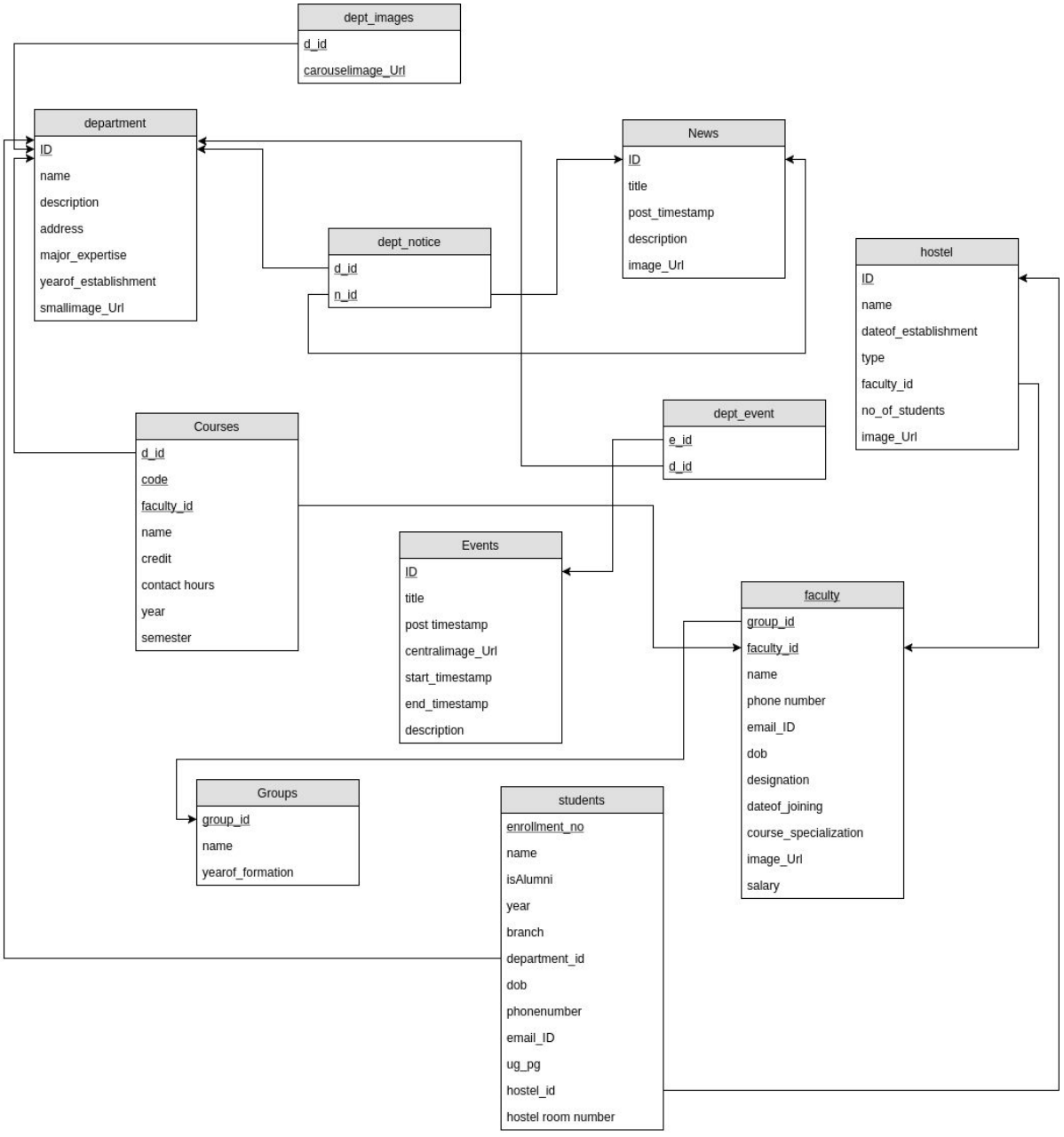
Event_images

{
e_ID varchar(20)
event_image Url varchar(100)
Primary key (e_ID,event_image)
Foreign key e_ID references Event
}

Hostel

{
ID varchar(20)
Name varchar(50)
date of establishment TIMESTAMP
Type varchar(10)
faculty Id varchar(20)
Number of students int(20)
Image Url varchar(100)
Primary key ID
Foreign key faculty id references Faculty
}

Final Schema Diagram :



2NF, 3NF, BCNF 4NF, 5NF?

Observation: In all our tables, we have FDs of type X -> A, X ∈ Candidate Key.

2NF:

Since X ∈ Candidate Key, no non-prime attribute can be dependent on a subset of candidate key.

3NF:

Since X ∈ Candidate Key, there cannot be a functional dependency from non-prime attribute to non-prime attribute

BCNF:

Since X ∈ Candidate Key ∈ Super Key, this means all our tables are in BCNF.

Since there are no multi-valued dependencies or Join Dependencies, the schema is in **5NF**

Introduction:

Our home page contains links to

1. Courses offered in the college
2. Departments available in the college
3. Current Events happening in the college
4. Internal College Groups
5. Student List with Data
6. Faculty Information
7. News happening on Campus

8. Hostels available

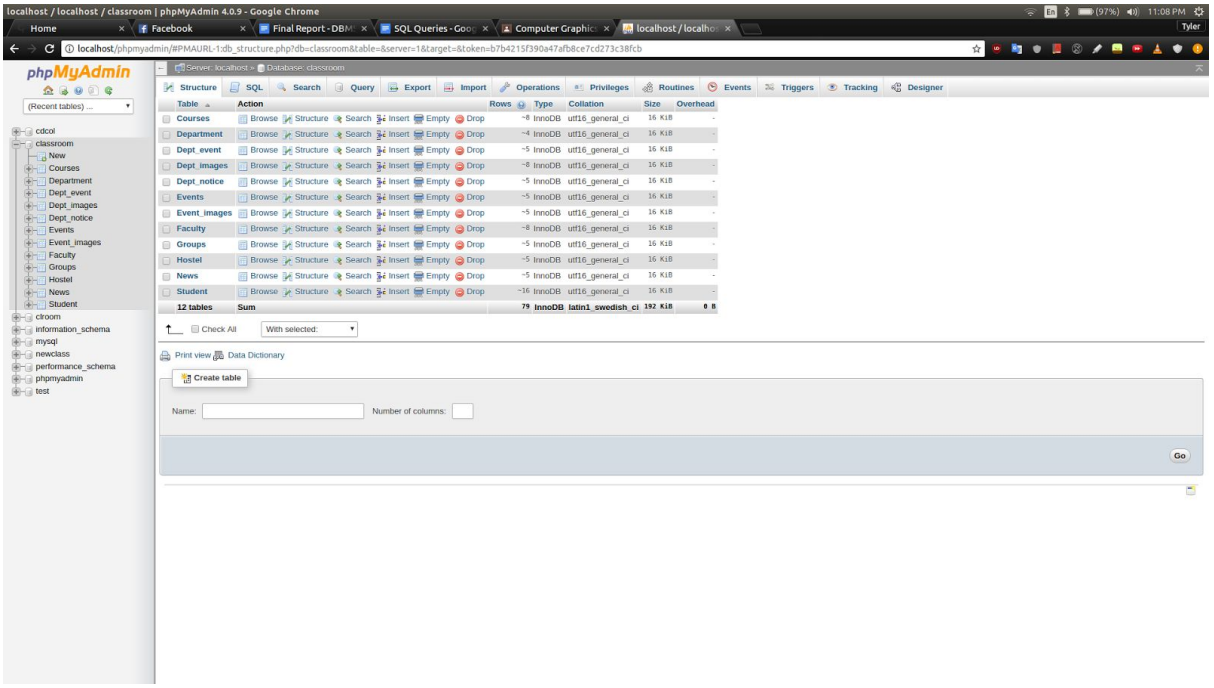
9. Contact information

There are separate pages for each of the departments of the college. Similarly there are separate pages for Events.

Final Database

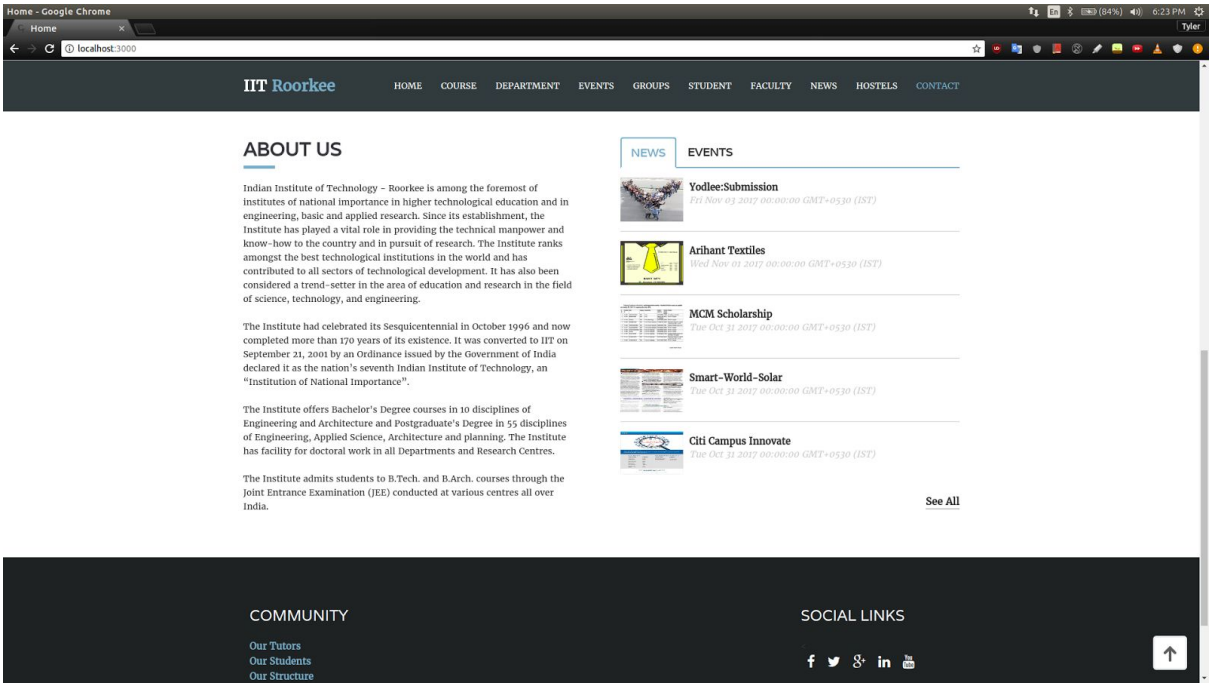
Some changes are made compared to our previous schema (as in part 1),

- From the Faculty relation, removed **Group_Id**
- Added attributes **Description** and **Faculty_Id** to the Groups relation.



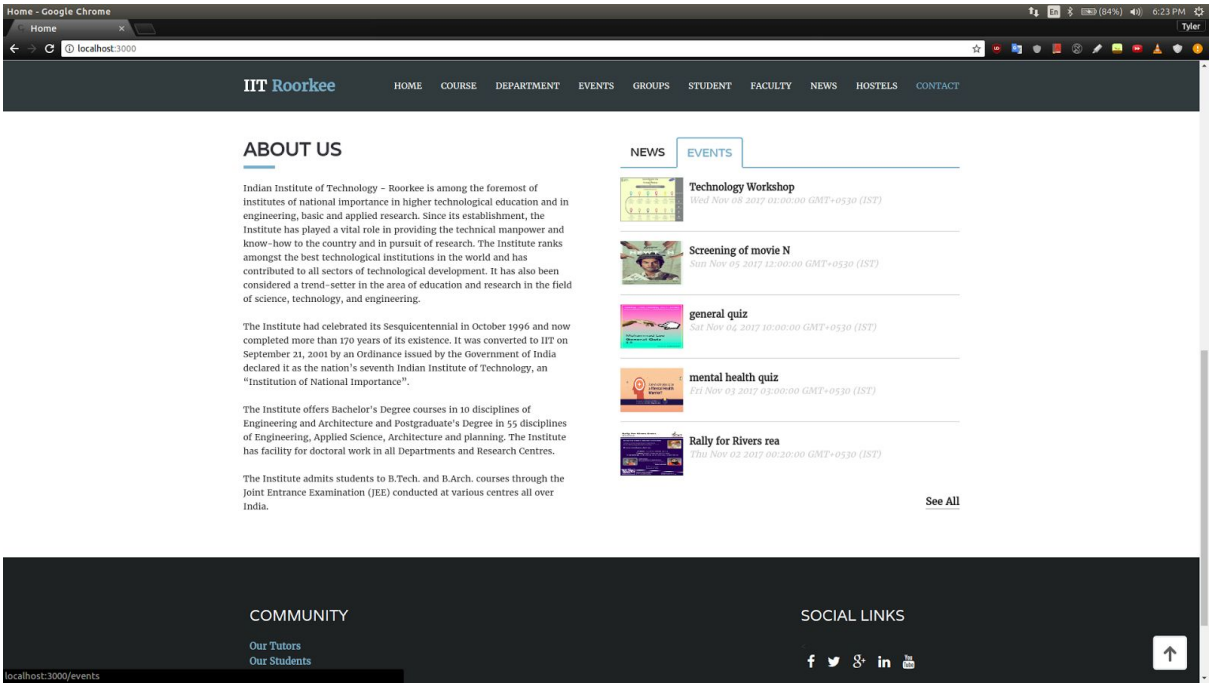
News

Current news in the campus



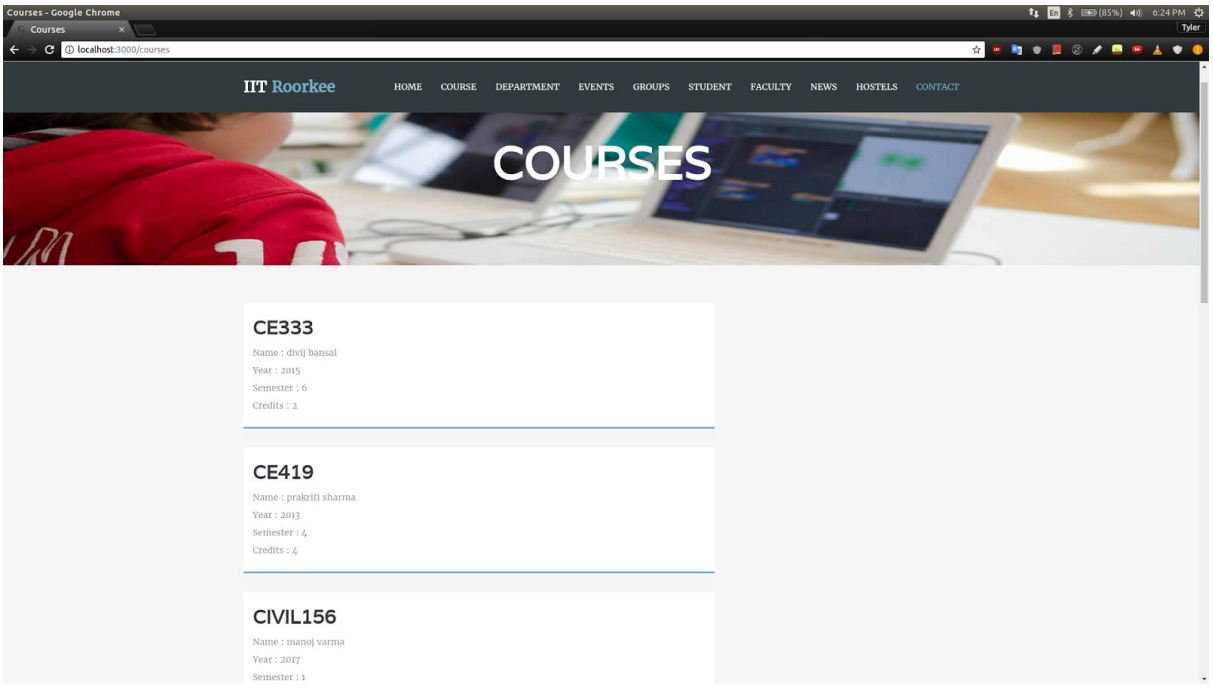
Events

Current events happening in the campus



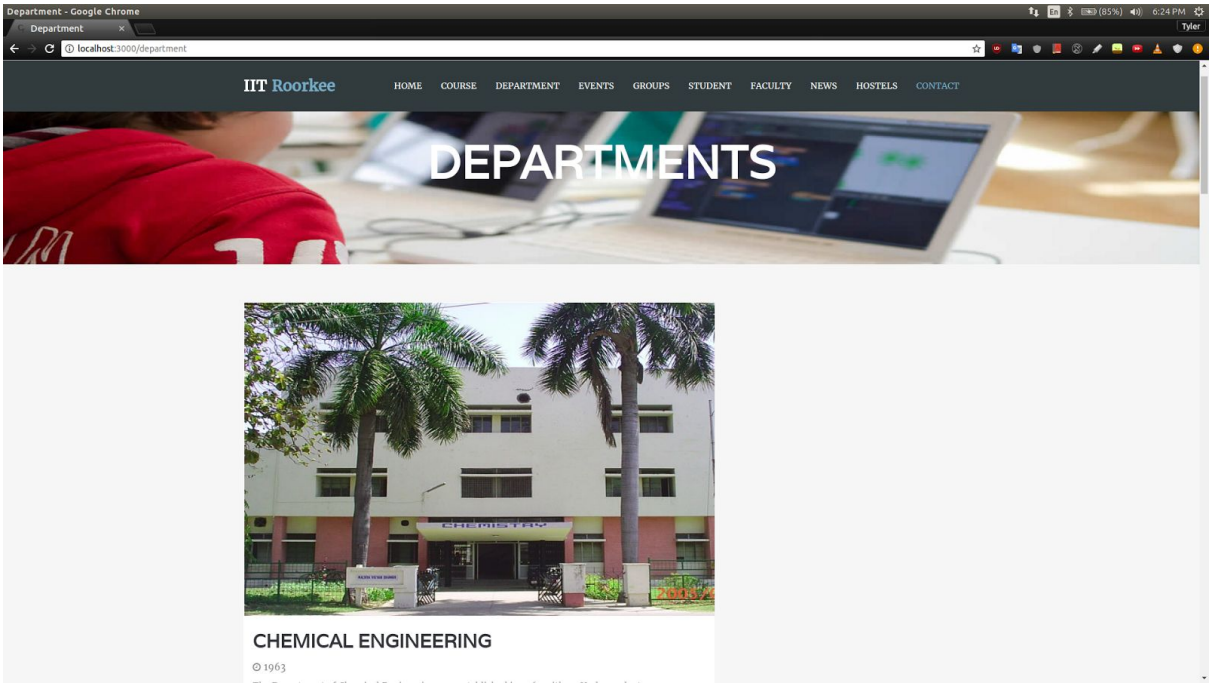
Courses

Courses offered in the college



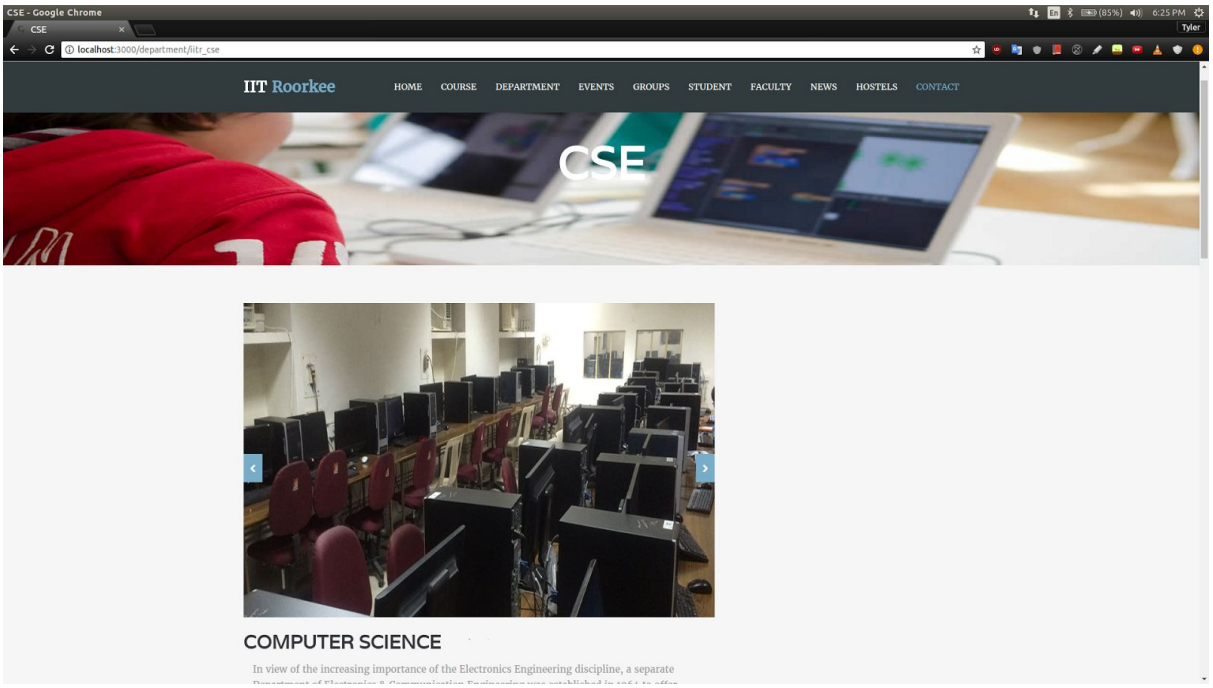
Departments as a whole

All the departments in the campus



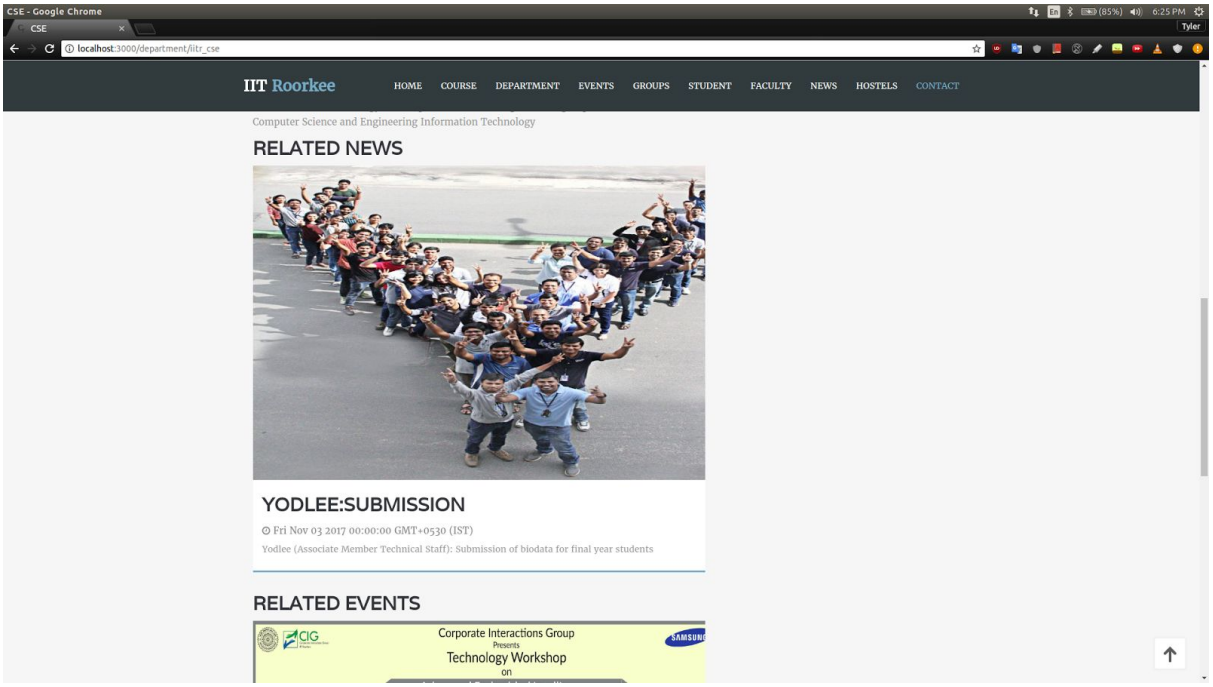
Departments Individually

More information about the departments



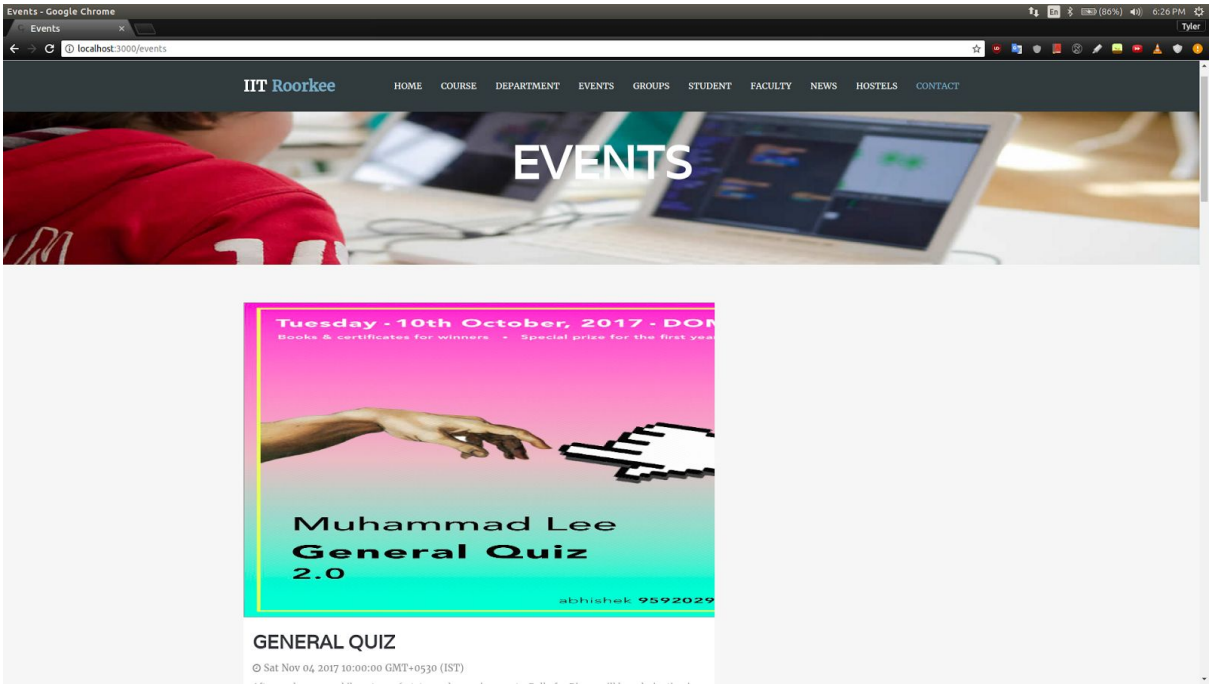
Related News and Events Department wise

All the news and events specific to the department are seen in this page



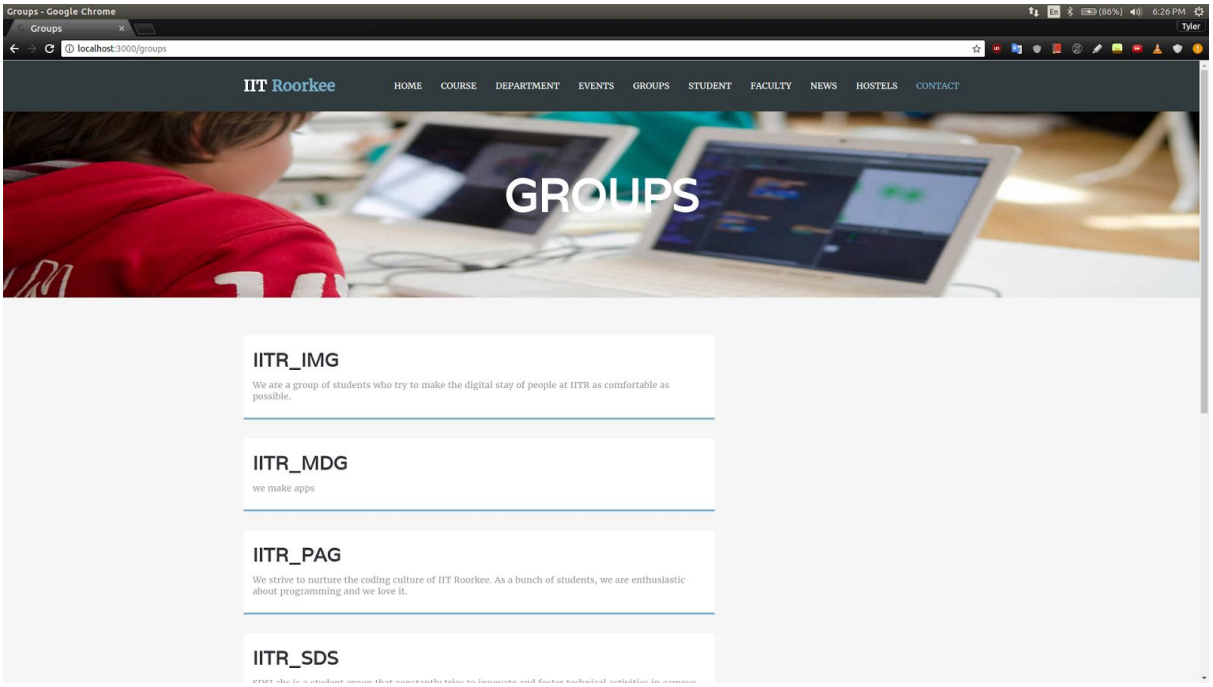
Separate Page for Events

All the events are written in a separate webpage



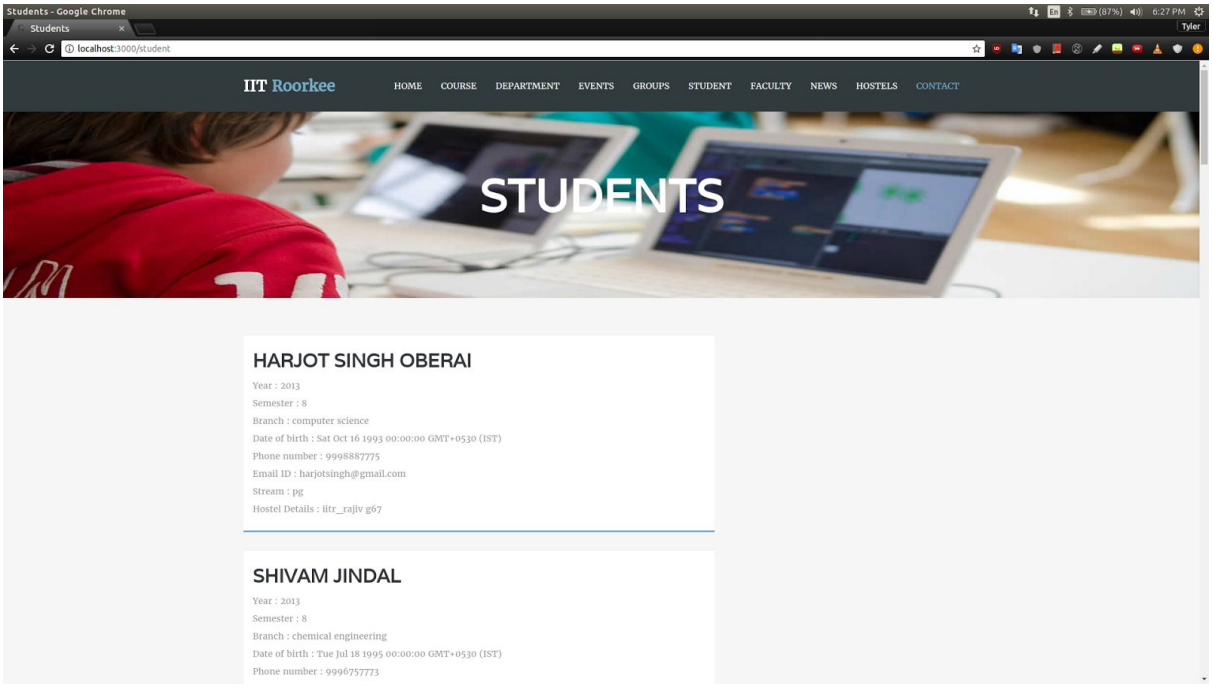
Campus Groups

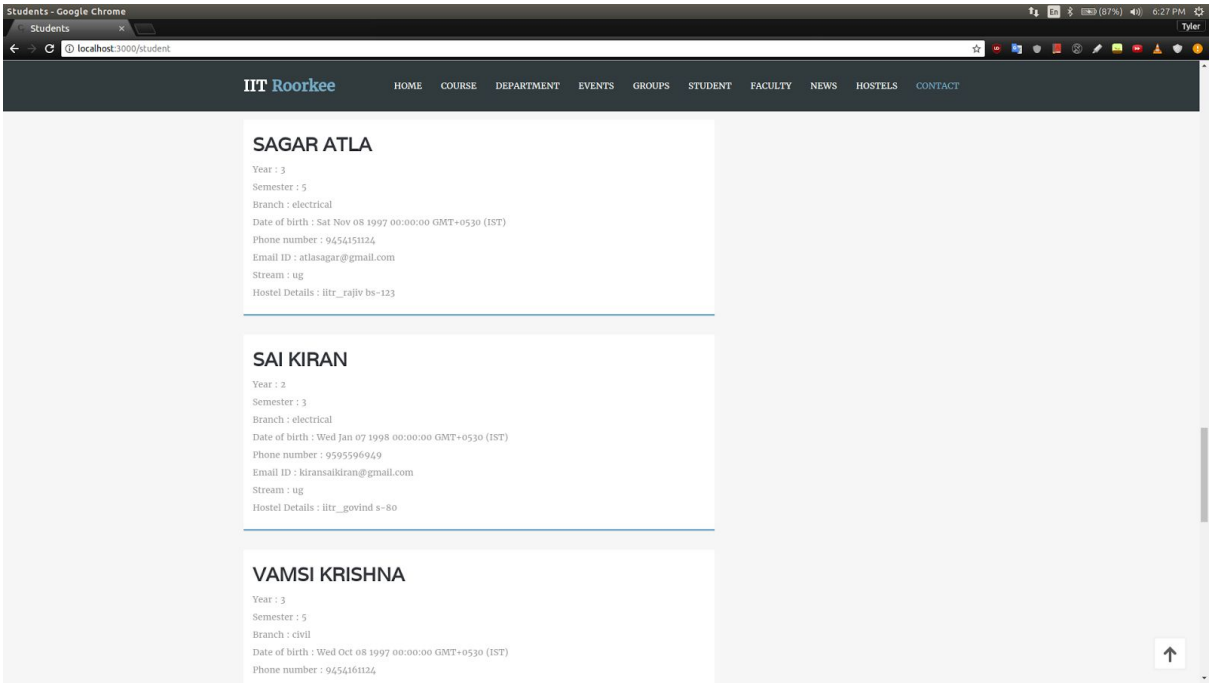
All the internal groups present in the campus



Students

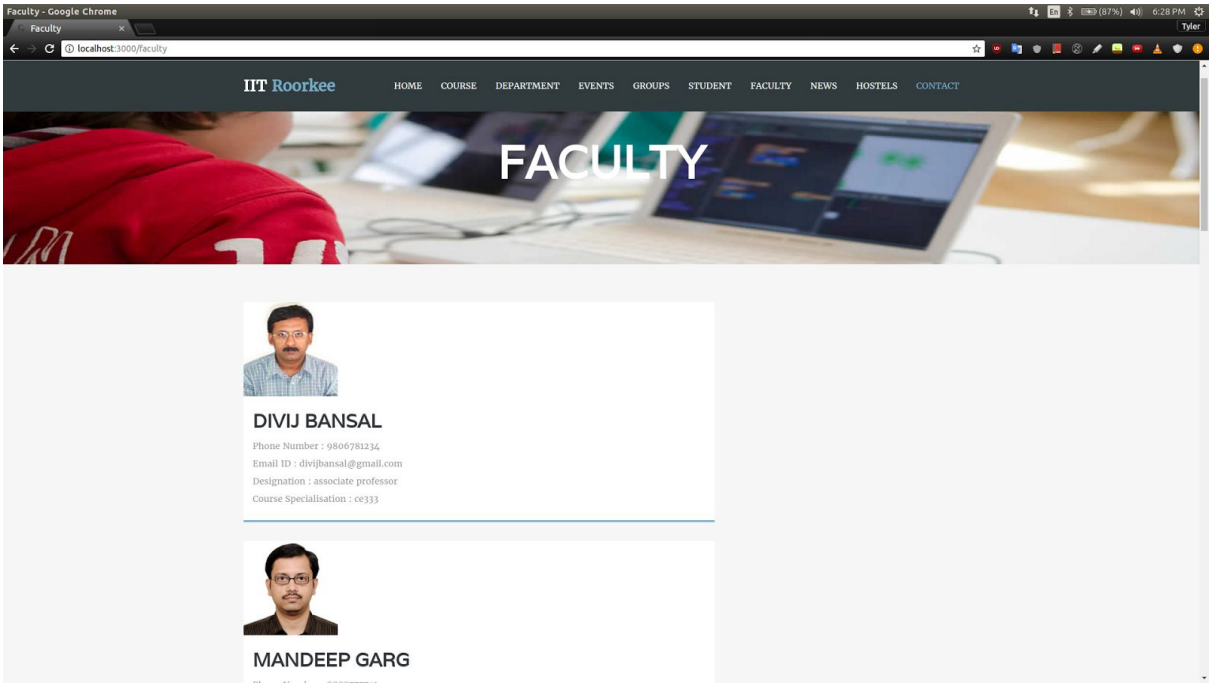
All the student info enrolled in the institute

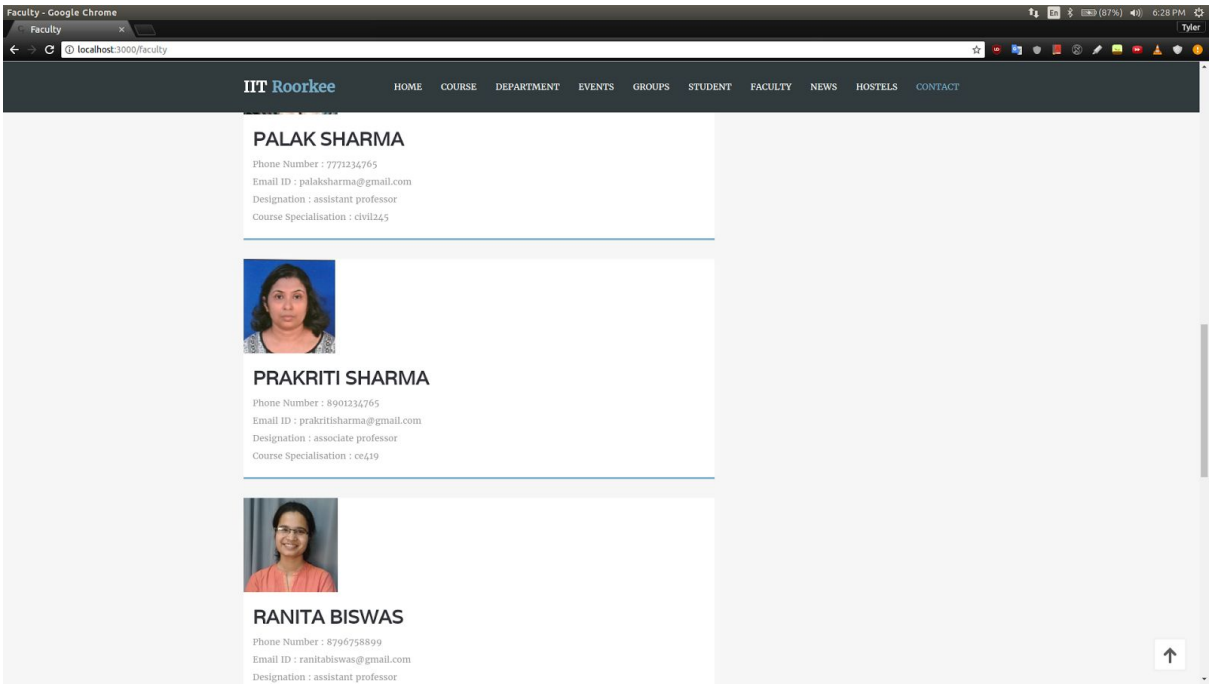




Faculty

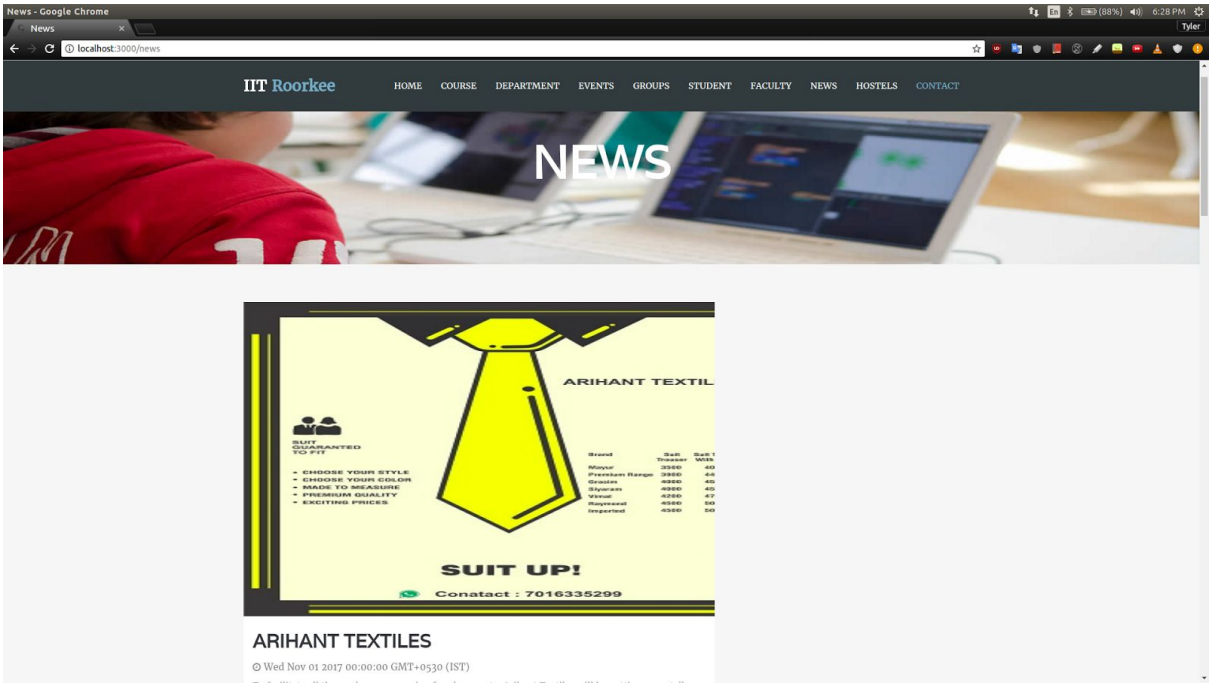
All the faculty working in the institute and the information about the same



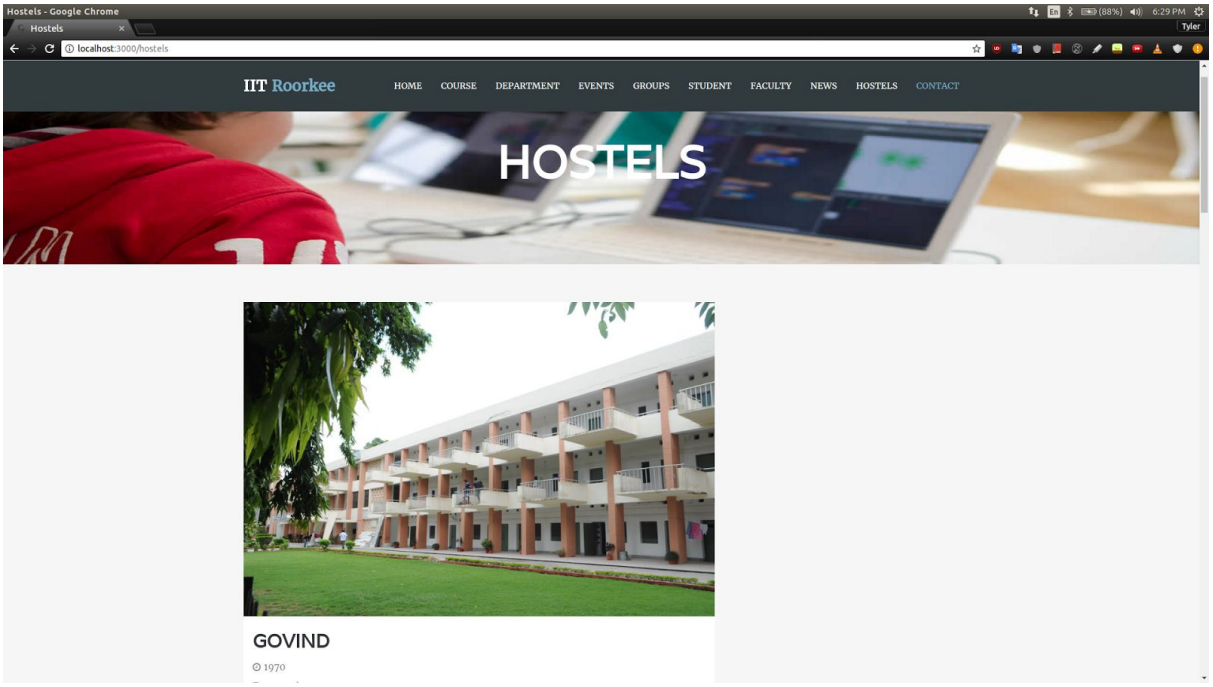


Separate Page for News

All the news about the campus are written in a separate webpage

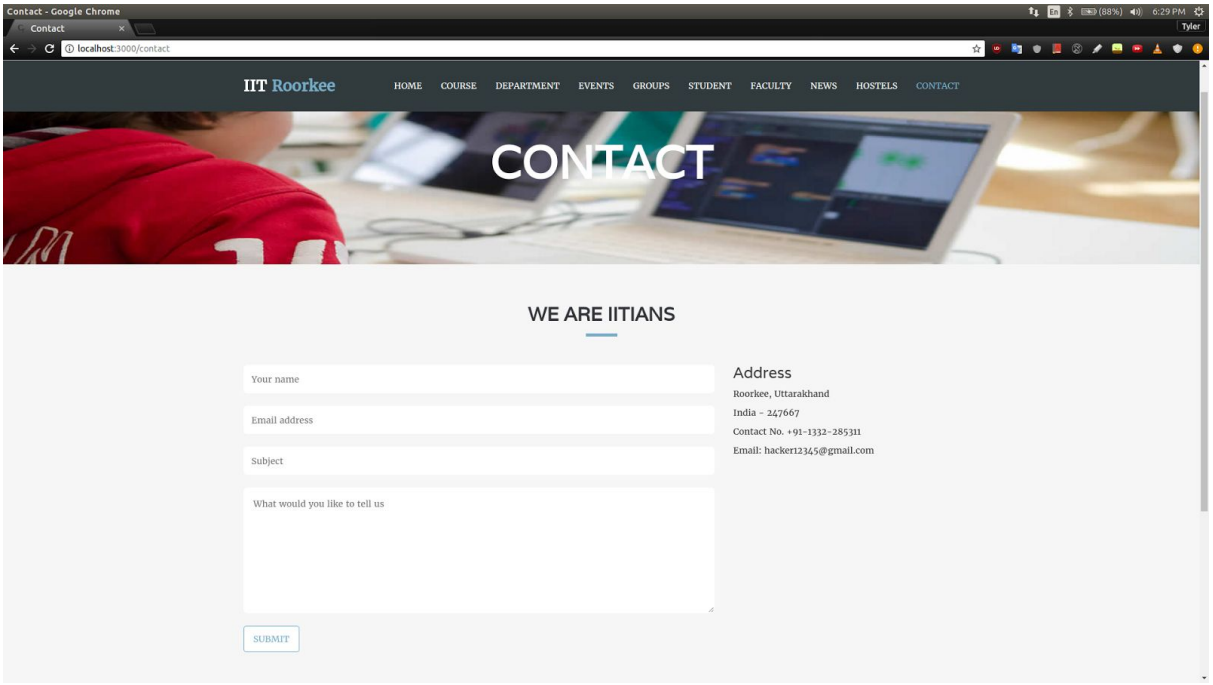


Hostels - All the available hostels in the institute



Contact Page

A separate webpage for contacting the institute authorities



Instructions:

- Install node.js
- Install npm in
- Install mysql-workbench

→ `sudo service mysql start`
→ `sudo service apache2 start`
→ `sudo mysql-workbench`

Clone the repository from github.
Cd to the directory
Import the database into the mysql workbench
Run the commands
→ `npm install`
→ `npm start`
Now the server will be running at “localhost:3000”

The extra features which we wanted to implement are the following -

1. For the the courses, we wanted to show the department , faculty and the students enrolled in that course . The queries for that goes as follows-

*Our database doesn’t have the data regarding the courses in which students are present..first that needs to be included. Let that be “course_data” which includes c_id - course id and s_id - student id .

-->SELECT * FROM Course,Course_data WHERE Course.ID=Course_data.c_id AND Course.ID = course id; - we will get the details of the students in the course from this

-->SELECT * FROM Course,Department WHERE Course.D_id=Department.ID AND Department.ID = departmentid of the course; - we will get the details of the department of course from this

-->SELECT * FROM Course,Faculty WHERE Course.faculty_ID=Faculty.ID AND Faculty.ID = faculty id of the course; - we will get the details of the faculty of the course from this

2. For the students, we wanted to show his department details and the courses he has enrolled in . The queries for that goes as follows-

-->SELECT * FROM Student,Department WHERE Student.department_id=Department.ID AND Department.ID = departmentid of the student; - we will get the details of the department of student from this

-->SELECT * FROM Course,Course_data WHERE Student.ID=Course_data.s_id AND Student.ID = student id; - we will get the details of the courses of the student from this

3. For the news, we wanted to create separate pages for it
   -->'SELECT * FROM News WHERE ID =?',[str] for the details of the news

4. For the faculty, we wanted to create a page for him and show his details along with the courses taught by him.
   -->'SELECT * FROM Faculty WHERE ID =?',[str]
   -->‘SELECT * FROM Faculty,Courses WHERE Faculty.ID=Courses.Faculty_ID AND Faculty.ID=?’,[str]